

REMARKS/ARGUMENTS

Claims 1-31 and 43-48 are pending. Claim 27 is amended and claim 42 is canceled.

Claims 1- 31 and 42-48 are rejected under 35 U.S.C. 102(e) as being anticipated by Bowman (US 6,725,399). Applicant submits that all of the claims currently pending in this application are patentably distinguishable over the cited references for the following reasons, and reconsideration and allowance of this application are respectfully requested.

The claimed invention is directed to a method for automatically preventing errors in computer software during its different life cycle phases. Some examples of different life cycle phases are depicted in FIG. 1, such as, Design phase, Development phase, Deploy phase, and Manage phase. According to the specification, the "system and method of the present invention use error detection as the catalyst for process improvement and error prevention. Each time a specific error is detected, the process is modified to prevent an entire class of errors. To ensure that this methodology becomes a long-lasting part of even the most rushed and chaotic software development processes, AEP includes technologies that automate as many software error prevention practices as possible, including historically difficult and time-consuming practices such as test case creation and verification of custom guidelines/requirements." (Page 5, lines 1-7, underlining added.). The claimed method and system of the present invention therefore "expose different types of mistakes and errors at the earliest possible phase of the software lifecycle." (Page 5, line 12.).

More particularly, the independent **claims 1 and 14** include, among other elements, "wherein each of the plurality of software verification tools corresponds to a respective lifecycle phase of the computer software, and automatically generates one or more test cases from the source code of the computer software," "generating verification results for each respective lifecycle phase of the computer software, responsive to executing the plurality of software verification tools and the automatically generated test cases," and "processing the verification results for generating a representation of functional behavior of the computer software." Bowman does not teach the above elements.

Rather, the system of Bowman performs requirement-based software testing "to determine if it [the software] will meet the specific requirements of an end user. . . . The method includes application in six key process areas. The six key process areas are development of a test plan, development of test cases to support the plan, development of an environment to simulate the technical environment in which the program will operate, test execution in which the tests are executed in a technical environment, compiling and analyzing the results and finally reporting the results in a form whereby the end user can determine both the feasibility of the software system for the specific requirements and any areas where additional testing or modifications are necessary. (See abstract, FIG. 1, and related text, emphasis added.). Therefore, the requirement-based testing method of Bowman is basically a "functional test" directed at the "end user," that is fundamentally different from the claimed invention that applies different verification tests to different lifecycle phases of the computer software.

First, Bowman does not disclose the element of "each of the plurality of software verification tools corresponds to a respective lifecycle phase of the computer software." Applicant respectfully disagrees with Examiner's characterization of the "Software Development/Implementation Process" of FIG. 1 of Bowman, as the claimed "lifecycle phases of the computer software." As explained above, the requirement-based software testing of Bowman applies only to the software development/implementation process." In deed, FIG. 1 clearly depicts that the "six key process areas" and the functional testing end when the software is released (see, "Release Date" arrow in FIG. 1). The cited "system test," "performance testing," runtime," and "acceptance testing" are all functional tests, which are performed before the software is released and do not include lifecycle phases of the computer software. In contrast, the claimed invention executes a plurality of software verification tools to verify a respective lifecycle phase of the software, for example, Design phase, Development phase, Deploy phase, and Manage phase, some of which are after the release.

Second, with respect to the element of each of the plurality of software verification tools "automatically generates one or more test cases from the source code of the computer software," Bowman does not teach this element either. Applicant respectfully disagree with the Examiner's

assertion that elements 2.7 [sic] and 2.12 of FIG. 3 of Bowman disclose the above limitation. Bowman clearly identifies FIG. 3 as representing the "test case development" process. Bowman continues with the emphasis that "test cases are created which support the test plan that was established by the first key process area. Additionally, test cases are designed to validate the software functions that that are in the software. The test cases are shaped by the test approaches defined in the test plan." (Col. 4, lines 38-45, underlining added.). Without a doubt, the terms used by Bowman to describe the "development," "creation," design," and "shaping" of the test cases do not teach neither explicitly, nor implicitly, automatic generation of test cases. In fact, FIG. 3 explains the cited box 2.7 as "Define/Refine Test Data." Box 2.3, which is clearly performed manually also uses the terms "Define/Refine" to describe the manual creation of test procedure. Therefore, according to Bowman, the "Define/Refine," "development," "creation," design," and "shaping" of the test cases is NOT the same as "automatically generat[ing] one or more test cases from the source code of the computer software," as required by claims 1 and 14. Nevertheless, Applicant would appreciate if the Examiner points to specific text of Bowman that may explicitly disclose automatic generation of test cases.

Third, regarding the element of "generating verification results for each respective lifecycle phase of the computer software," Bowman does not teach this, because, as explained above, Bowman's software development/implementation process of FIG. 1 is not the same as lifecycle phase of the software, for example, Design phase, Development phase, Deploy phase, and Manage phase, some of which are after the release.

Consequently, for at least each of the above three reasons, Bowman does not teach all elements of **claims 1 and 14** and therefore, claims 1 and 14 are not anticipated by Bowman.

Independent **claims 28 and 45** include, among other elements, "providing a known error in the computer software, the known error belonging to a class of errors," "providing a plurality of software verification tools each of the plurality of software verification tools corresponding to a respective lifecycle phase of the computer software," "analyzing the known error in the computer software to determine what phase of the lifecycle the error was introduce," and "customizing a verification scope of one or more of the plurality of verification tools that

correspond to the lifecycle phase that the known error was introduced." Bowman does not teach the above elements.

First, regarding the element of "providing a known error in the computer software, the known error belonging to a class of errors," there is no teaching of providing a know error in Bowman. Rather, Bowman objective is to "not only [can] determine how well a product has been tested by the vendor and/or creator, but rather if the product will reliably meet the user requirements in performing specific functions, and the quality of the performance." (Col. 2, lines 35-38.). No known error is provided in any of the "six key process areas," or any of the tests performed by Bowman's method.

Second, the element of "providing a plurality of software verification tools each of the plurality of software verification tools corresponding to a respective lifecycle phase of the computer software," is not disclosed by Bowman, as explained above.

Third, the element of "analyzing the known error in the computer software to determine what phase of the lifecycle the error was introduce," is also not disclosed by Bowman. As discussed above, with respect to claims 1 and 14, Bowman's software development/implementation process of FIG. 1 is not the same as lifecycle phase of the software, for example, Design phase, Development phase, Deploy phase, and Manage phase, some of which are after the release.

Fourth, with respect to "customizing a verification scope of one or more of the plurality of verification tools that correspond to the lifecycle phase that the known error was introduced," Bowman does not teach i) verification scope for each of the verification tools, ii) customization of the verification scope, or finally iii) determining what phase of the lifecycle the error was introduce.

As a result, for at least each of the above four reasons, Bowman does not teach all elements of **claims 28 and 45** and thus, claims 28 and 45 are not anticipated by Bowman either.


Dependent claims 2-13, 15-26, 28-31, 42-44, and 46-48 are dependent from allowable independent claims 1, 14, 27, and 45, respectively and therefore include all the limitations of their base claims and additional limitations therein. Accordingly, these claims are also allowable

Appln No. 10/613,166
Amdt date August 3, 2007
Reply to Office action of May 18, 2007

over the cited references, as being dependent from an allowable independent claim and for the additional limitations they include therein.

In view of the foregoing amendments and remarks, it is respectfully submitted that this application is now in condition for allowance, and accordingly, reconsideration and allowance are respectfully requested.

Respectfully submitted,
CHRISTIE, PARKER & HALE, LLP

By 
Raymond R. Tabandeh
Reg. No. 43,945
626/795-9900

RRT/clv

CLV PAS748960.1-*08/3/07 2:56 PM